Hacia la Solución del Problema de Chequeo de Instancia en Ontologías Inconsistentes Usando Argumentación Rebatible

Sergio Alejandro Gómez, Carlos Iván Chesñevar, Guillermo Ricardo Simari

Laboratorio de Investigación y Desarrollo en Inteligencia Artificial (LIDIA)[1]
Depto. Cs. e Ing. de la Computación – Universidad Nacional del Sur
Av. Alem 1253 - (8000) Bahía Blanca - Argentina
Tel/Fax: (+54) 291-4595135/6
E-mail: {sag, cic, grs}@cs.uns.edu.ar

Resumen

La *Web Semántica* es una visión futura de la Web en la cual la información tiene un significado exacto definido en términos de *ontologías*. Esto permitirá que agentes inteligentes procesen el contenido de los recursos web. Las definiciones de ontologías se realizan en el lenguaje OWL, el cual es una notación XML para sentencias expresadas en Lógicas para la Descripción. Cuando una ontología es inconsistente, los razonadores actuales no son capaces de obtener conclusiones a partir la misma.

En esta línea de investigación, exploramos la utilización de la argumentación rebatible para obtener respuestas significativas al problema de la determinación de la pertenencia de individuos a una clase en presencia de ontologías inconsistentes en el marco de la Web Semántica. La propuesta de trabajo consiste en transformar una ontología expresada en una Lógica para la Descripción en un Programa Lógico Rebatible. Bajo ciertas condiciones, en el caso en que la ontología sea inconsistente, se podrá determinar la pertenencia de un cierto individuo a una clase particular utilizando un análisis dialéctico que considerará las razones a favor y en contra de la pertenencia de dicho individuo a esa clase. En este artículo comentamos brevemente los resultados que hemos obtenido en la solución de dicho problema y discutimos los problemas que restan resolverse.

Palabras clave: Inteligencia Artificial, Razonamiento no-monótono, Argumentación rebatible, Web Semántica, Ontologías, Lógicas para la Descripción, Agentes inteligentes

1. Introducción y motivaciones

La *Web Semántica* [2] es una visión futura de la Web en la cual la información tiene un significado exacto; permitiendo así que las computadoras entiendan y razonen en base a la información encontrada en la Web. En la Web Semántica se propone resolver el problema de la asignación de semántica a los recursos web por medio de *definiciones de ontologías*.

Las definiciones de ontologías se realizan en el lenguaje OWL (por Ontology Web Language) [9], el cual está basado en las Lógicas para la Descripción (DL) [1]. Las ontologías expresadas en las DL son definidas en términos de un conjunto de *axiomas terminológicos* (TBox) conteniendo información de clases y un conjunto de *axiomas asercionales* (ABox) conteniendo información sobre individuos perteneciendo a las clases definidas en la Tbox. Si bien las ontologías *consistentes* expresadas en DL pueden ser manipuladas por razonadores DL (*e.g.*, Racer [8]), debido a que las ontologías son entes complejos, creados por uno o varios ingenieros de conocimiento, surgen *inconsistencias*. Una manera posible de trabajo consiste en *reparar* la ontología inconsistente para que los razonadores DL puedan

[1]LIDIA es un miembro del IICyTI (Instituto de Investigación en Ciencia y Tecnología Informática).

trabajar con ellas. Otro enfoque consiste en *aceptar* inconsistencias y lidiar con ellas por medio de un formalismo de razonamiento no monótono.

En particular, la *Programación en Lógica Rebatible* (DeLP) [4] provee un lenguaje para la representación de conocimiento y el razonamiento que utiliza la *argumentación rebatible* [3, 10, 11] para decidir entre conclusiones contradictorias a través de un *análisis dialéctico*. La codificación de conocimiento por medio de un programa DeLP provee un buen balance entre expresividad e implementabilidad, que permite lidiar con información incompleta y potencialmente contradictoria.

En esta línea de investigación, exploramos la utilización de la argumentación rebatible para obtener respuestas significativas al problema de la determinación de la pertenencia de individuos a una clase en presencia de ontologías inconsistentes en el marco de la Web Semántica. La propuesta de trabajo consiste en transformar una ontología expresada en una Lógica para la Descripción en un Programa Lógico Rebatible. Bajo ciertas condiciones, en el caso en que la ontología sea inconsistente, se podrá determinar la pertenencia de un cierto individuo a una clase particular utilizando un análisis dialéctico que considerará las razones a favor y en contra de la pertenencia de dicho individuo a esa clase. En este artículo comentamos brevemente los resultados que hemos obtenido en la solución de dicho problema y discutimos los problemas que restan resolverse.

El resto de artículo está estructurado de la siguiente manera. En la Sección 2 se introducen los fundamentos de las Lógicas para la Descripción y la Programación en Lógica Rebatible. En la Sección 3 se repasan brevemente los resultados obtenidos hasta el momento. Finalmente, en la Sección 4 se discuten los problemas abiertos y las soluciones factibles.

2. Marco de trabajo

2.1. Lógicas para la Descripción

Las *Lógicas para la Descripción* (DL) [1] son una familia de formalismos de representación de conocimiento basados en las nociones de *conceptos* (predicados unarios, clases) y *roles* (relaciones binarias), y están principalmente caracterizados por constructores que permiten describir conceptos complejos y roles a partir de otros atómicos usando conjunción ($C \sqcap D$), disyunción ($C \sqcup D$), negación ($\neg C$), restricción existencial ($\exists R.C$), y restricción de valor ($\forall R.C$).

Una ontología $\mathcal{O} = (T, A)$ consiste de dos conjuntos finitos y mutuamente disjuntos. El conjunto T (Tbox) introduce la *terminología* y el conjunto A (Abox) contiene *aserciones* acerca de objetos particulares en el dominio de aplicación. Las sentencias de la Tbox son de la forma $C \sqsubseteq D$ (*inclusiones*) y $C \equiv D$ (*igualdades*), donde C y D son descripciones de conceptos (posiblemente complejas).

Es posible dar una semántica a las ontologías al considerar la correspondencia existente entre las DL y la Lógica de Primer Orden (FOL) [1]. Básicamente, los conceptos (roles) DL se corresponden con predicados unarios (binarios, resp.) y los operadores $\neg$, $\sqcap$, $\sqcup$ con negaciones, conjunciones y disyunciones lógicas clausuradas universalmente. Las expresiones de la forma $\exists R.C$ se entienden como expresiones FOL de la forma $(\forall x)(\exists y)(R(x, y) \rightarrow C(y))$. Por otro lado, las inclusiones $C \sqsubseteq D$ se corresponden con expresiones de la forma $(\forall x)(C(x) \rightarrow D(x))$ y las igualdades $C \equiv D$ son una abreviación de dos inclusiones de la forma $C \sqsubseteq D$ y $D \sqsubseteq C$.

2.2. Argumentación rebatible

En un programa lógico rebatible $\mathcal{P} = (\Pi, \Delta)$, se pueden distinguir un conjunto Δ de reglas rebatibles $P \prec Q_1, \ldots, Q_n$, y un conjunto Π de reglas estrictas $P \leftarrow Q_1, \ldots, Q_n$. La derivación de literales en DeLP resulta en la construcción de argumentos. Un *argumento* $\langle \mathcal{A}, H \rangle$ es un conjunto no contradictorio y minimal de cláusulas fijas $\mathcal{A}$ de Δ que permite derivar un literal fijo H posiblemente usando reglas fijas de Π.

Debido a que los argumentos pueden estar en conflicto (concepto capturado en términos de una contradicción lógica) puede definirse una relación de ataque entre argumentos. Usualmente se define un criterio para decidir entre dos argumentos en conflicto. Si el argumento atacante es estrictamente preferido sobre el argumento atacado, entonces éste es llamado un *derrotador propio*; en cambio, si no hay comparación posible, entonces éste es llamado un *derrotador por bloqueo*.

Para determinar si un argumento dado $\mathcal{A}$ es considerado finalmente no derrotado (o *garantizado*), se lleva a cabo un proceso dialéctico recursivo, en el cual son tomados en cuenta los derrotadores para $\mathcal{A}$, los derrotadores de éstos y así sucesivamente. Dado un programa DeLP $\mathcal{P}$ y una consulta H, la respuesta final a H con respecto a $\mathcal{P}$ toma en cuenta tal análisis dialéctico. La respuesta a una consulta puede ser: SÍ, NO, INDECISO, o DESCONOCIDO.

3. Resultados obtenidos

En trabajos previos [5, 6], mostramos cómo una ontología definida en Lógicas para la Descripción puede ser expresada como un programa DeLP. Dada una ontología $\mathcal{O} = (T, A)$, ésta es expresada como un programa rebatible $\mathcal{P} = (\Pi, \Delta)$. El conjunto de axiomas de la terminología T es expresado como un conjunto de reglas rebatibles Δ. El conjunto de aserciones sobre individuos es codificado como un conjunto de hechos Π.

Explicaremos brevemente la idea detrás de la traducción de Lógicas para la Descripción a DeLP. Sea $\mathcal{L}_{DL}$ el lenguaje de las DL y sea $\mathcal{L}_{DeLP}$ el lenguaje de los programas DeLP. La función de traducción $\mathcal{T} : \mathcal{L}_{DL} \rightarrow \mathcal{L}_{DeLP}$ toma sentencias de la Abox y por cada una retorna un hecho; por cada aserción de inclusión de la Tbox de la forma $C \sqsubseteq D$ retorna una regla rebatible de la forma $D(X) \prec C(X)$. Esta función se basa en la estudiada por Grosof *et. al.* [7] y se fundamenta en el hecho de que las sentencias $Tbox$ de inclusión $C \sqsubseteq D$ pueden ser identificadas con fórmulas de lógica de primer orden de la forma $(\forall x)(C(x) \rightarrow D(x))$. Así, debido a que estas fórmulas en realidad pueden ser de complejidad arbitraria, bajo ciertas condiciones pueden ser reescritas como programas lógicos. A continuación mostramos un breve ejemplo.

Ejemplo 3.1 *Consideremos la siguiente ontología $\mathcal{O} = (T, A)$ tal que:*

$$T = \left\{ \begin{array}{l} boxer \sqsubseteq violent \\ boxer \sqcap animal\_lover \sqsubseteq \neg violent \\ \exists loves.airedale \sqsubseteq animal\_lover \\ \exists kicks.poodle \sqsubseteq \neg animal\_lover \end{array} \right\}, \text{ y } A = \left\{ \begin{array}{l} boxer(john) \\ loves(john, lola) \\ airedale(lola) \\ kicks(john, tina) \\ poodle(tina) \end{array} \right\}$$

La Tbox T expresa que: los boxeadores son violentos; un boxeador que es amante de los animales no es violento; si alguien ama a un perro de raza Airedale Terrier es un amante de los animales, y, si alguien es capaz de patear a un Poodle, entonces no es un amante de los animales. La Abox A expresa que: John es boxeador y ama a Lola; Lola es un Airedale Terrier; John patea a Tina y Tina es un Poodle.

La ontología presentada en el ejemplo anterior es claramente inconsistente ya que desde el punto de los razonadores DL estándar la clase de los boxeadores debería ser vacía. Esto se debe a que es posible demostrar que un boxeador es violento y no violento al mismo tiempo (*i.e.*, $boxer \sqsubseteq violent \sqcap \neg violent = \bot$), y por otro lado se conoce un boxeador llamado John (*i.e.*, $boxer(john)$).

Nuestra propuesta de trabajo consiste en expresar la ontología original $\mathcal{O} = (T, A)$ como un programa DeLP $\mathcal{P} = (\Pi, \Delta)$ donde $\Pi = \mathcal{T}(A)$ y $\Delta = \mathcal{T}(T)$. Una vez hecho, la pertenencia de un individuo a a una clase C se resuelve realizando un análisis dialéctico para determinar si existe un argumento garantizado para $\langle \mathcal{A}, C(a) \rangle$ (a pertenece a C) o para $\langle \mathcal{B}, \sim C(a) \rangle$ (a no pertenece a C).

Ejemplo 3.2 (Continúa el Ejemplo 3.1) *Consideremos el programa* $\mathcal{P} = (\Pi, \Delta)$ *que se obtiene al traducir a DeLP a la ontología* $\mathcal{O} = (T, A)$*:*

$$\Pi = \left\{ \begin{array}{l} boxer(john) \\ loves(john, lola) \\ airedale(lola) \\ kicks(john, tina) \\ poodle(tina) \end{array} \right\}, \text{ y}$$

$$\Delta = \left\{ \begin{array}{l} violent(X) \prec boxer(X) \\ \sim violent(X) \prec boxer(X), animal\_lover(X) \\ animal\_lover(X) \prec loves(X, Y), airedale(Y) \\ \sim animal\_lover(X) \prec kicks(X, Y), poodle(Y) \end{array} \right\}$$

Para determinar si John pertenece la clase de los violentos, es necesario determinar si existe un argumento garantizado a favor del literal $violent(john)$*.*

Así, encontramos un argumento $\langle \mathcal{A}_1, violent(john) \rangle$ *a favor de que John es violento, donde:*

$$\mathcal{A}_1 = \{ \; violent(john) \prec boxer(john) \; \}$$

Pero hay otro argumento que derrota al primero, indicando que John no es violento: $\langle \mathcal{A}_2, \sim violent(john) \rangle$*, con:*

$$\mathcal{A}_2 = \left\{ \begin{array}{l} \sim violent(john) \prec boxer(john), animal\_lover(john); \\ animal\_lover(john) \prec loves(john, lola), airedale(lola) \end{array} \right\}$$

Finalmente, aparece un tercer argumento $\langle \mathcal{A}_3, \sim animal\_lover(john) \rangle$ *socavando la conclusión anterior en el punto de ataque* $animal\_lover(john)$*, y reinstaurando al primero:*

$$\mathcal{A}_3 = \{ \; \sim animal\_lover(john) \prec kicks(john, tina), poodle(tina) \; \}$$

Así, como el resultado a las consultas $violent(john)$ *es* SI *y* $\sim violent(john)$ *es* NO*, concluimos que John pertenece a la clase de los violentos.*

4. Trabajo en progreso

Se debe notar que en el acercamiento presentado en la sección anterior, todas las sentencias de la Tbox original son expresadas como reglas rebatibles en el programa DeLP resultante de la transformación. Esto se puede interpretar como que todas las inclusiones de conjuntos expresadas por la Tbox son consideradas como reglas *default*.

En ciertas aplicaciones, es interesante que ciertas sentencias de la Tbox sean interpretadas como reglas estrictas y otras como reglas rebatibles. Formalmente, dada una ontología $\mathcal{O} = (T, A)$, nos interesará particionar al conjunto T en un conjunto de sentencias *estrictas* T_S y otro conjunto de sentencias *rebatibles* T_D, tal que $T_S \cup T_D = T$ y $T_S \cap T_D = \emptyset$. Así, el programa DeLP obtenido aplicando la transformación sería de la forma: $\mathcal{P} = (\mathcal{T}(T_S) \cup \mathcal{T}(A), \mathcal{T}(T_D))$.

Además, una vez que tenemos información estricta es necesario considerar la deducción de información negativa. Por ejemplo, la sentencia de inclusión $C \sqsubseteq D$ (todos los C's son D's) es traducida como la regla estricta $D(X) \leftarrow C(X)$. Claramente, a partir de $C(a)$ (a pertenece al concepto C) podemos deducir $D(a)$ (a pertenece al concepto D). Sin embargo, en presencia de $\sim D(a)$ (a no pertenece a D) no es posible deducir $\sim C(a)$ (a no pertenece a C). En virtud de ello, dada una sentencia DL $C_1 \sqcap C_2 \sqcap \ldots \sqcap C_{n-1} \sqcap C_n \sqsubseteq D$, en lugar de sólo incluir la única regla estricta $D(X) \leftarrow C_1(X), C_2(X), \ldots, C_{n-1}(X), C_n(X)$ en el programa DeLP derivado de la ontología, una solución que se presenta prometedora consiste en incluir a todas las *transpuestas* de dicha regla, *i.e.*:

$$
\begin{array}{l}
D(X) \leftarrow C_1(X), C_2(X), \ldots, C_{n-1}, C_n(X) \\
\sim C_1(X) \leftarrow \sim D(X), C_2(X), \ldots, C_{n-1}, C_n(X) \\
\sim C_2(X) \leftarrow \sim D(X), C_1(X), \ldots, C_{n-1}, C_n(X) \\
\ldots \\
\sim C_{n-1}(X) \leftarrow \sim D(X), C_1(X), C_2(X), \ldots, C_n(X) \\
\sim C_n(X) \leftarrow \sim D(X), C_1(X), C_2(X), \ldots, C_{n-1}(X)
\end{array}
$$

Nuestro trabajo de investigación actual está orientado a la formalización de las soluciones a los problemas planteados en esta sección así como a realizar una caracterización teórica del proceso.

Agradecimientos

Esta investigación está fundada por Agencia Nacional de Promoción Científica y Tecnológica (PICT 2002 No. 13.096, PICT 2003 No. 15.043, PAV 2004 076), por CONICET (Argentina), por Proyecto TIN2006-15662-C02-02.

Referencias

[1] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider, editors. *The Description Logic Handbook – Theory, Implementation and Applications*. Cambridge University Press, 2003.

[2] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scient. American*, 2001.

[3] C. Chesñevar, A. Maguitman, and R. Loui. Logical Models of Argument. *ACM Computing Surveys*, 32(4):337–383, December 2000.

[4] A. García and G. Simari. Defeasible Logic Programming: An Argumentative Approach. *Theory and Prac. of Logic Program.*, 4(1):95–138, 2004.

[5] Sergio Alejandro Gómez, Carlos Iván Chesñevar, and Guillermo Ricardo Simari. An Approach to Handling Inconsistent Ontology Definitions based on the Translation of Description Logics into Defeasible Logic Programming. In *Proc. of the XII Congreso Argentino de Ciencias de la Computación (CACIC'06)*, pages 1185–1196, 2006.

[6] Sergio Alejandro Gómez, Carlos Iván Chesñevar, and Guillermo Ricardo Simari. Inconsistent Ontology Handling by Translating Description Logics into Defeasible Logic Programming. *Revista Iberoamericana de Inteligencia Artificial*, 11(35):11–22, 2007.

[7] Benjamin Grosof, Ian Horrocks, Raphael Volz, and Stefan Decker. Description Logic Programs: Combining Logic Programs with Description Logics. *WWW2003, May 20-24, Budapest, Hungary*, 2003.

[8] Volker Haarslev and Ralf Möller. RACER System Description. Technical report, University of Hamburg, Computer Science Department, 2001.

[9] Deborah L. McGuiness and Frank van Harmelen. OWL Web Ontology Language Overview, 2004. http://www.w3.org/TR/owl-features/.

[10] Henry Prakken and Gerard Vreeswijk. Logical Systems for Defeasible Argumentation. In D. Gabbay and F.Guenther, editors, *Handbook of Philosophical Logic*, pages 219–318. Kluwer Academic Publishers, 2002.

[11] G. Simari and R. Loui. A Mathematical Treatment of Defeasible Reasoning and its Implementation. *Artificial Intelligence*, 53:125–157, 1992.